Amendments to the Claims:

This listing of claims will replace all prior versions, and listings of claims in the application:

Listing of Claims:

1. (Previously presented) A scalable processing system having general purpose registers and a general purpose memory, comprising:

a memory device that is operative to store a plurality of executable program instructions in definable locations, each said executable program instruction being propagated through said memory device, wherein position of each of said executable program instructions in said memory device is associated with a current timetag and each said timetag is indicative of the nominal sequential order of execution of said associated executable program instructions;

a plurality of processing elements distributed throughout said memory device, each of said processing elements being configured and arranged to receive executable program instructions from current positions of said memory device, wherein each of said processing elements executes said executable program instructions associated with said current position without regard to order; and

a plurality of active stations associated with each of said processing elements for enforcing programmatic ordering of said executable program instructions as indicated by the state of said timetag.

2. (Currently amended) The scalable processing system ~~method~~ according to claim 1 wherein said program instructions are propagated in a linear sequence.

3. (Currently amended) The scalable processing system ~~method~~ according to claim 1 wherein said program instructions are propagated in a block.

4. (Currently amended) The scalable processing system ~~method~~ according to claim 1 wherein said program instructions are propagated in columns.

5. (Previously presented) A scalable processing system having general purpose registers and a general purpose memory, comprising:

a memory device that is operative to store a plurality of executable program instructions in definable locations, each said executable program instruction being propagated through said memory device, wherein position of each of said executable program instructions in said memory device is associated with a current timetag and each said timetag is indicative of the nominal sequential order of execution of said associated executable program instructions;

a plurality of processing elements distributed throughout said memory device, each of said processing elements being configured and arranged to receive executable program instructions from current positions of said memory device, wherein each of said processing elements executes said executable program instructions associated with said current positions without regard to order having the highest priority;

a plurality of active stations associated with each of said processing elements for enforcing programmatic ordering of said executable program instructions as indicated by the state of said timetag data field, each of said active stations having a dedicated timetag register for capturing a temporally closest previous broadcast timetag value for comparison with a timetag of a datum sharing a common address in the active station in order to accomplish at least one of the following: a) to enforce said programmatic ordering, b) to link said instruction with a closest previous related instruction as indicated by a common memory address in said general purpose random access memory, a predicate address, or a register address in said general purpose register, or c) to minimize dependencies among instructions.

6. (Previously presented) A scalable general purpose processing system for assuring correct processing of instructions according to a resource flow execution model, said system comprising:

a plurality of sharing groups;

a plurality of processing elements, each processing element being associated with at least one sharing group, each processing element operative to generate an output result as a result of executing a program instruction;

a plurality of spanning buses of uniform preselected segment length arranged in columns, said spanning buses being of a column height which is independent of bus length;

a forwarding unit associated with each sharing group, each said forwarding unit being operative to store an output result from a processing element, each said forwarding unit in each column being coupled to monitor its corresponding level in an adjacent spanning bus segment;

a memory device coupled to said spanning buses operative to store a plurality of executable program instructions, wherein each of said executable program instructions includes a timetag data field indicative of the nominal sequential order of said associated executable program instructions; and

a plurality of active stations, each of said active stations for holding a single one of said program instructions, said plurality of active stations forming one said sharing group, and each said forwarding unit in a column being coupled to each said sharing group in said column,

each of said processing elements being configured to receive said executable program instructions from said memory device, wherein each of said processing elements is operative to execute any of said executable program instructions as soon as its operand is acquired thereby generating an output result, whereby a plurality of executable program instructions are executed in parallel during each instruction cycle.

7. (Previously presented) The processing system according to claim 6 wherein said spanning buses are arranged in end-to-end fashion.

8. (Previously presented) The processing system according to claim 6 wherein said timetag data field is limited in length in order to permit reuse of values.

9. (Previously presented) The processing system according to claim 7 wherein said spanning bus columns are coupled in a loop.

10. (Previously presented) A scalable processing system, comprising:

a memory device that is operative to store a plurality of executable program instructions in definable locations, each said executable program instruction being propagated through said memory device;

a plurality of processing elements, each of said processing elements associated with an output result that is produced as a result of executing one of said executable program instructions;

a plurality of active stations, each of said active stations for holding a single one of said program instructions, each of said active stations being associated with at least one of said processing elements for execution of said single program instruction; and

a spanning bus structure configured to couple at least some output results associated with said processing elements to at least some of said active stations, each output result having a timetag and an address associated with it,

each of said active stations comprising at least one input, said input having a timetag and an address associated with it, said input being associated with execution of said single program instruction,

each active station further comprising:

first comparison logic operative to compare said timetag associated with its input with said timetag associated with an output result on said spanning bus and to produce a first comparison result;

second comparison logic operative to compare said address associated with its input with an address associated with an output result on said spanning bus and to produce a second comparison result; and firing logic operative to issue said single program instruction to one of said processing elements multiple times, each time being based on said first comparison result and said second comparison result.

11. (Previously presented) The processing system of claim 10, wherein said spanning bus is further configured to couple outputs of one or more active stations to other active stations.

12. (Previously presented) The processing system of claim 10, wherein said timetags are indicative of a nominal sequential order of execution of said associated executable program instructions.

13. (Previously presented) The processing system of claim 10, wherein said active station further comprises third comparison logic operative to compare a value of said input to a value of an output result on said spanning bus and to produce a third comparison result, said firing logic further operative to issue said program instruction to one of said processing elements multiple times, each time being based on said first comparison result, said second comparison result, and said third comparison result.

14. (Previously presented) The processing system of claim 10 wherein said input is a datum that is used as an operand by said single program instruction.

15. (Previously presented) The processing system of claim 14 wherein said datum is content of a register and said address associated with said process input is information that identifies said register.

16. (Previously presented) The processing system of claim 10 wherein said input is a predicate that is used by a processing element to determine whether said single program instruction is executed.

17. (New): In a processing system, a method of executing loop instructions comprising machine instructions which constitute a loop, said loop instructions including a reverse branch instruction, the method comprising:

storing a first copy of said loop instructions in an execution window, including modifying said reverse branch instruction in said first copy of said loop instructions to produce a forward branch instruction; and

storing at least a second copy of said loop instructions in said execution window, including modifying a branch address of said reverse branch instruction in said second copy to said address of said first instruction in said first copy of said loop instructions.

18. (New): The method of claim 17 wherein said reverse branch instruction is a conditional branch instruction, wherein said first step of modifying includes reversing the sense of a predicate evaluation of the conditional branch instruction.

19. (New): The method of claim 17 further comprising prior to said first step of storing, making a determination whether said execution window can contain said first and second copies of said loop instructions.

20. (New): In a processing system, a method of executing program instructions comprising:

loading a plurality of instructions into an execution window, each instruction having one or more instruction inputs and producing an instruction output upon execution thereof;

concurrently executing instructions in at least a subset of said instructions that are loaded in said execution window irrespective of data dependencies and control dependencies among said subset of instructions, to produce a plurality of first instruction outputs;

broadcasting at least a subset of said first instruction outputs to instruction inputs of at least some of said instructions in said instruction window; and

selectively re-executing said instructions in said subset of instructions if their respective instruction inputs have changed as a result of said broadcasting step.

21. (New): The method of claim 20 wherein said subset of instructions includes data instructions which produce data instruction outputs from data instruction inputs upon execution.

22. (New): The method of claim 20 wherein said subset of instructions includes predicate instructions which produce predicate instruction outputs from predicate instruction inputs upon execution.

23. (New): The method of claim 20 further comprising re-executing an instruction more than once.

24. (New): In a processing system, a method of executing program instructions comprising:

concurrently executing instructions in a plurality of first program instructions absent a step of enforcing data dependencies and control dependencies among said first instructions, to produce a plurality of first instruction outputs, wherein some of said first instruction outputs are instruction inputs to some of said first program instructions;

detecting a change in one or more instruction inputs of said first program instructions as a result of production of said first instruction outputs; and

selectively re-executing those of said first program instructions whose instruction inputs have changed.

25. (New): The method of claim 24 wherein one or more instruction outputs serve as an instruction input to one of said first program instructions, wherein said first program instruction is re-executed if a sequence order of execution of one of said instruction outputs is earlier in time than a sequence order of execution of said first program instruction and is the closest in time to said first program instruction than others of said instruction output.

26. (New): The method of claim 24 wherein said first program instructions are stored in a memory, wherein a sequence order of execution of said first program instructions is determined according to an order in which said first program instructions are stored in said memory.

27. (New): A scalable processing system having general purpose registers and a general purpose memory, comprising:

a memory device that is operative to store a plurality of executable program instructions in definable locations, each said executable program instruction being propagated through said memory device;

a plurality of processing elements distributed throughout said memory device, each of said processing elements being configured and arranged to receive executable program instructions from current positions of said memory device, wherein said processing elements executes said executable program instructions associated with said current position irrespective of data dependencies and control dependencies among said executable program instructions, thus producing program instruction outputs; and

broadcast logic operative to broadcast said program instruction outputs as program instruction inputs to at least some of said executable program instructions, wherein said executable program instructions are selectively re-executed if their respective program instruction inputs have changed by said program instructions outputs.

28. (New): The system of claim 27 wherein said executable program instructions includes data instructions which produce data instruction outputs from data instruction inputs upon execution.

29. (New): The system of claim 27 wherein said executable program instructions includes predicate instructions which produce predicate instruction outputs from predicate instruction inputs upon execution.

30. (New): The system of claim 27 further comprising re-executing an executable program instruction more than once.

31. (New): In a processing system, a method for executing program instructions comprising:

when a branch instruction is encountered, concurrently executing first program instructions prior to said branch instruction and second program instructions between said branch instruction and a target location identified by a branch address associated with said branch instruction;

while executing said first program instructions, storing first instruction outputs produced by said execution, each first instruction output having an associated register address;

while executing said second program instructions, storing second instruction outputs produced by said execution, each second instruction output having an associated register address; and

if a branch-taken condition occurs, then for a second instruction output that has a register address equal to that of a first instruction output and which was produced later in time, then replacing said second instruction output with said first instruction output, wherein said branch-taken condition arises when a predicate evaluation indicates that execution of said second program instructions should not be performed.

32. (New): The method of claim 31 wherein said first instruction output is a data output or a predicate output and said second instruction output is a data output or a predicate output.

33. (New): A scalable processing system comprising:

an instruction window operative to store a plurality of executable program instructions, said executable program instructions including at least one branch instruction thereby defining a first branch path and at least a second branch path; and

a memory device that is operative to store a plurality of executable program instructions in definable locations, and is operative to perform multipath execution of said first branch path and said second branch path wherein execution of said first branch path and said second branch path occur concurrently,

said memory device further operative to concurrently execute first executable program instructions comprising said first branch path irrespective of data dependencies and control dependencies among said first executable program instructions to produce a plurality of first instruction outputs, said memory device further operative to broadcast at least a subset of said first instruction outputs to instruction inputs of at least some of said first executable program instructions and to selectively re-execute said first executable program instructions if their respective instruction inputs have changed as a result of broadcasting said first instruction outputs,

said memory device further operative to concurrently execute second executable program instructions comprising said second branch path irrespective of data dependencies and control dependencies among said second executable program instructions, to produce a plurality of second instruction outputs, said memory device further operative to broadcast at least a subset of said second instruction outputs to instruction inputs of at least some of said second executable program instructions and to selectively re-execute said second executable program instructions if their respective instruction inputs have changed as a result of broadcasting said second instruction outputs.